Software-Defined Mobile Backhaul for Future Train to Ground Communication Services

Aravinthan Gopalasingham, Quan Pham Van, Laurent Roullet, Chung Shue Chen
Eric Renault, Lionel Natarianni, Stephane De Marchi, Emmanuel Hamman
Nokia Bell Labs, Route de Villejust, 91620 Nozay, France
SAMOVAR, Telecom SudParis, CNRS, Universite Paris-Saclay, Evry, France
Simpulse, 7 rue de la Croix Martre, 91120 Palaiseau, France
Emails: {gopalasingham.aravinthan, laurent.roullet, chung_shue.chen, lionel.natarianni}@nokia.com
Emails: {eric.renault, quan.pham-van}@telecom-sudparis.eu, {stephane.demarchi, emmanuel.hamman}@simpulse.fr

***Abstract*—Software Defined Networking (SDN) has attracted tremendous interest in the telecommunication industry due to its ability to abstract, manage and dynamically re-configure end-to-end networks from a centralized controller. Though SDN is considered to be a suitable candidate for various use cases in mobile networks, none of the work so far has discussed its advantages and actual realization for Train-to-Wayside Communication System (TWC). In this paper, for the first time, the architecture and use cases of SDN controlled mobile backhauling framework for TWC is proposed. We discuss how our proposed architecture can efficiently handle mobility management and also provide dynamic quality-of-service (QoS) for different services on board. As a first step, a software prototype is developed using industrial standard OpenDayLight SDN controller to have our architecture evaluated. Since the automotive sector is being considered to be an important driver for 5G network, our SDN based mobile backhauling solution can be positioned in 5G where SDN plays an important role.**

***Index Terms*—SDN, Intelligent Transport System, TWC, LTE/5G, OpenDaylight, OpenFlow, Open vSwitch.**

I. Introduction

In recent years, the rail industry has been migrating its communication networks from voice centric 2G systems with limited data transmission capabilities to IP based 4G technologies to enable high speed Internet access for passengers on board and future intelligent transport systems (ITS) [1]. Since traditional metro railway communication services are built on non-IP based network, it has been a difficult task to integrate new IP based applications and services. In the early stage of this technology migration, the railway operator deployed WLAN to support the communication networks and services such as the communications-based train control (CBTC) and closed-circuit television video (CCTV) surveillance systems. However, the lack of QoS features in WiFi based solution limits the WLAN scheme to carry in parallel both mission-critical CBTC traffic and non-mission critical CCTV traffic in the same network properly. Note that WiMAX was considered in many TWC deployments to deliver best-effort passenger Internet service until LTE took over WiMAX due to its far stronger ecosystem [2]. Although both WLAN and LTE are based on all-IP architecture, the advantages of LTE including lower user-plane latency, support for sophisticated QoS, guaranteed traffic delivery through multi service network, and end-to-end IP architecture have attracted metro and rail operators to choose LTE as the single technology to deploy their communication networks [1]–[3].

Although it is possible to provide high speed communication services, the rail industry still has many unsolved issues such as packet loss due to frequent handover, difficulty in providing guaranteed QoS for mission critical services due to train's mobility, interference coming from high vibration and thermal challenging environment, and limitation to wireless infrastructure due to tunnels [4]. Besides, the excessive multimedia applications from smart phones also generate heavy traffic that can affect the QoS of mission critical services. Despite the fact that it is difficult to define a unique solution for all the above technical issues, by exploiting the advantages of recent SDN technologies there is an opportunity to address some of the above mentioned issues by defining a re-configurable wireless backhauling network that can deliver guaranteed QoS for mission critical services and also handle frequent handover virtually inside the backhaul network rather than in the mobile core network as in LTE.

During the last few years, SDN has significantly impacted the telecom industry due to the advantage of being able to dynamically and intelligently program network equipments from the centralized controller in order to enforce networks to serve according to user and service requirements. For example, OpenFlow is a widely accepted and standardized SDN protocol for providing end-to-end QoS for different services with respect to traffic variations over time, change in network topology, etc [5]. SDN is to allow service providers to increase the efficiency of their networks and fine tune their existing infrastructure more dynamically. Though there is significant research and experiment work on deploying SDN/OpenFlow in mobile networks (see for example [6] and references therein), none of them so far has discussed or shown how to integrate SDN into the train to ground communication networks. Since automotive sector (ITS) is an important use case for 5G where SDN is a key technology to bring high level of abstraction and dynamic programmability in the underlying architecture, positioning SDN in TWC systems is not only a solution to address various mentioned problems but also an opportunity for future 5G integration.

Inspired by the advantages of SDN and also considering the possible road map of next generation mobile networks

(NGMN), we propose in this work a novel SDN based mobile backhaul framework for future train to ground communication services addressing issues related to mobility management and QoS. In our scheme, the mobile backhaul network is completely re-configurable via the SDN controller to offer guaranteed QoS for delay sensitive services. Our solution is a way to evolve from the traditional concept of *Network-as-an-Asset* to the concept of *Network-as-a-Service* [7]. We developed a software prototype using OpenDayLight (ODL) [8], an industrial standard SDN controller, software OpenFlow switches and Open vSwitch(OVS) [9], Mongo DB [10] and emulated wireless modems. We also validated our design using user-attachment and handover scenarios deployed as North Bound (NB) applications.

The rest of the paper is organized as follows. Section II discusses the state-of-the-art of TWC and research on SDN related. Section III presents the architecture and use cases of software defined mobile backhaul network for TWC services. Section IV describes our software prototype to validate the architecture. Finally, some discussions, future direction and conclusion are given in Section V.

II. State of the Art

A. Evolution of Train-to-Ground Communication Systems

The early train-to-ground communication systems such as Private Mobile Radio (PMR) were operated using Terrestrial Trunked Radio (TETRA) to carry mainly voice operations and narrowband data services [11]. TETRA is operated at Ultra High Frequency (UHF) in the range of 420-470 MHz band. However, due to the limitation in bandwidth and the excessive use of UHF spectrum, PMR has migrated to higher bandwidth digital system. Global System for Mobile Communications - Railway (GSM-R), which is a standard based on GSM, was introduced by European Rail Traffic Management System (ERTMS) to carry both signaling information and voice communication. GSM-R system has many advantages including the support for fast handover with train speed up to 500 km/h, which is an important development for TWC [1].

Clearly, today's TWC is not only for deploying services related to train operation and management but also for providing real time safety and security for passengers on board (CCTV) and value added services such as multimedia entertainment, passenger travel information, etc. In addition, the growth of Internet and the introduction of smart devices (smart phones, tablets, etc) necessitates the availability for seamless Internet connections to users independent of their locations. Currently, there are several solutions based on recent technologies such as WLAN, WiMAX, Satellite Link and LTE, for supporting high speed TWC networks [4]. Among all the solutions, LTE has an immense potential in terms of high uplink and downlink data rate along with end-to-end QoS for mission critical CBTC traffic and real time surveillance systems [2].

B. SDN for Wireless Communication Systems

The concept of SDN has attracted widely the research communities and industries after the introduction of OpenFlow, the first standardized SDN protocol as a way to experiment new protocols and applications [5]. OpenFlow laid the possibility of using its intrinsic feature of flow based traffic treatment to achieve traffic engineering, traffic monitoring, load balancing, provision of end-to-end QoS, and network virtualization, in various applications such as transport networks, data centers, local area network (LAN), wide area network (WAN), etc [12].

Initially, SDN drew its attention towards offering switching and routing solutions for fixed networks. Recently there has been many research works on aligning SDN to wireless networks. OpenRoads is the first research work analyzing the capability of OpenFlow for mobile networks [13]. SoftRAN proposed an architecture of SDN controller that abstracted the radio access network (RAN) for coordinated scheduling, interference management and load balancing [14]. Generalized SDN Platform for RAN is demonstrated in [15] based on the architecture of ODL to support different use cases both in distributed and centralized RANs [16].

The Software Defined Wireless Transport Networks (SDWTNs) [17] analyzed programmability in wireless transport networks. Results claimed that traffic shaping and dynamic spectrum allocation can be achieved using SDN based wireless transport networks in future 5G systems. Our work is also inspired by the concept of SDWTNs, but we analyzed the advantages of using SDN in the context of TWC and for the first time presented the architecture and provided use cases of a novel SDN controlled mobile backhauling framework for train to ground communication networks.

III. Software Defined Mobile Backhaul

A. Architecture Overview

Our architecture is defined using SDN principle to bring programmability in mobile backhauling of TWC. The main challenge is to design a controller that provides NB APIs for applications to control/re-configure backhaul network in real time and to integrate wireless devices to operate under SDN. Our solution consists of different components including SDN controller, Software Defined wireless modems, and OpenFlow supported traffic Aggregation Switch for connecting to Internet Gateway and mobile core network, as shown in Fig. 1. There are two kinds of software defined wireless modems in our architecture: (i) Infra Modem (IM), a fixed modem in the ground connected to the service provider networks, and (ii) User Modem (UM), a mobile modem placed in the train connected to ground network via IM. UM is attached to different service nodes (WiFi, LTE Femto, CCTV, CBTC, etc) through LAN in the train. Each IM can simultaneously serve multiple UMs.

Since our framework can provide service level QoS, we place train control systems (CCTV, CBCT, etc) and Passenger Information Systems (PIS) in the global Internet rather than as separate entity as in traditional metro train operations. Due to the advantage of high uplink/downlink data rate for mobile broadband services, we back on LTE as a technology for providing mobile phone connectivity in the train. The passengers on board can use LTE femto to have seamless mobile

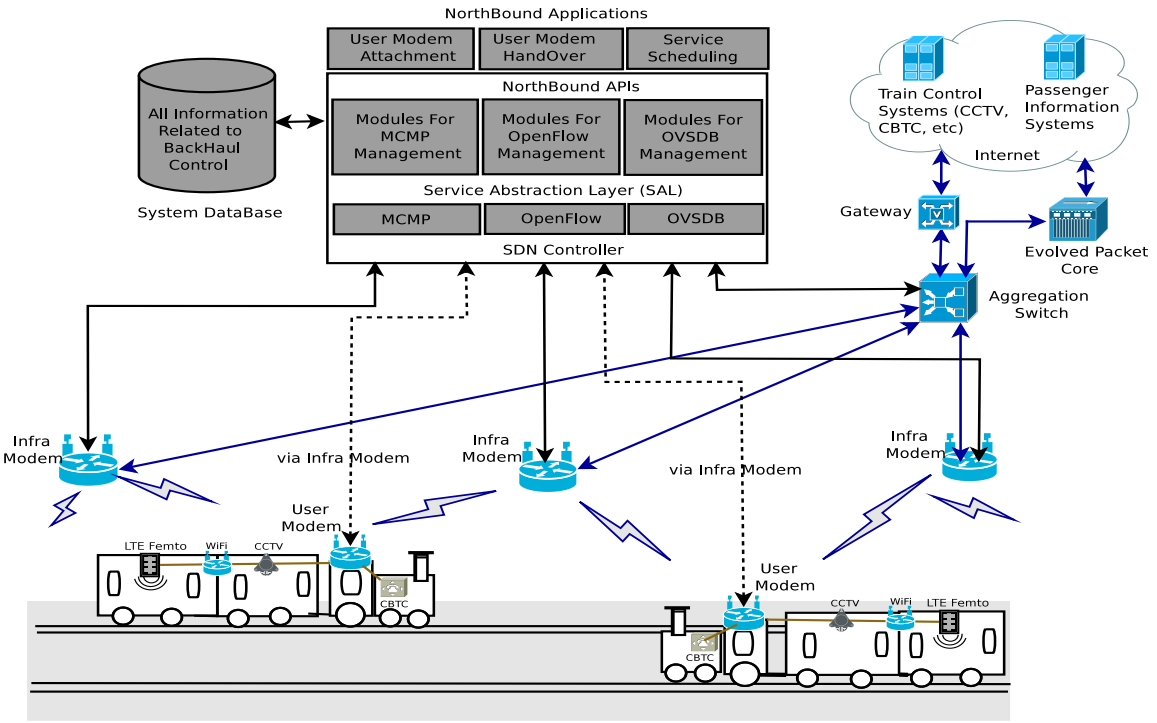


Fig. 1: Software Defined Mobile Backhauling for TWC Architecture.

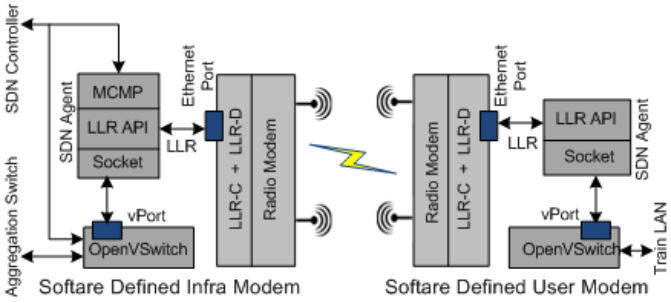


Fig. 2: Software Defined Radio Modem Architecture.

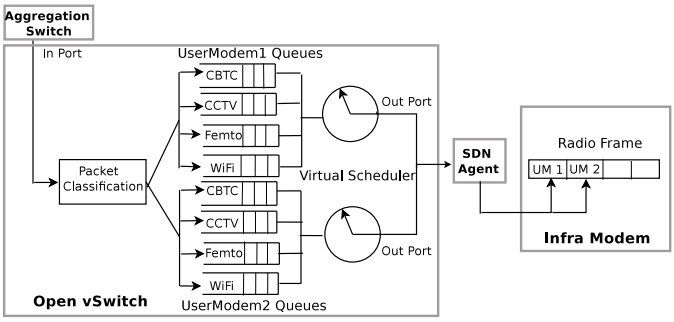


Fig. 3: Packet Scheduling using Priority Queuing in OVS.

connectivity as well as WiFi hotspots for Internet access. Taking the benefits from QoS provision in OpenFlow [18], our software defined wireless backhaul can dynamically adopt QoS for different traffic flows. This is necessary for certain services in train, for example CBTC is the most time critical service in the train which needs accurate information about train location in order to efficiently manage the operations from the control room.

Our controller design closely follows the architecture from our recent work [15], where we used external database (DB) together with ODL controller as a way to store measurements and configuration details to be used by NB applications. Modem Control and Management Protocol (MCMP) is the South Bound (SB) protocol we introduced in addition to standard OpenFlow (OF) and Open vSwitch Database Management Protocol (OVSDB) [19] to handle messages related to mobility management procedures such as User Modem Attachment and User Modem Handover. These messages are originated from the Low Level Radio Control (LLR-C) messages between IM and UM. MCMP is defined using ZeroMQ (ZMQ) [20] due to high performance and ability to establish many-to-many connection between endpoints, this is necessary because the controller needs to handle request/response from multiple Infra Modems in parallel to avoid latency in control decisions. NB applications use MCMP for sending and receiving messages to/from IM for mobility management. MCMP management modules in the controller interact with system DB to maintain mobility related information to be used by applications. OF and OVSDB protocols are used by applications to interact with OVS for establishing QoS aware service scheduling in both IM and UM.

B. Software Defined Radio Modem

Radio Modems such as IM and UM in our architecture is an embedded Software Defined Radio (SDR) modems

designed using pulsar boards [21]. Pulsar has a scalable and programmable hardware architecture that includes variable field-programmable gate arrays (FPGAs) for medium-to-high performance data needs and can integrate 2×2 Radio Frequency (RF) Transceivers to support bandwidths from 5 to 20 MHz at carrier frequency from 60 MHz to 6 GHz. It has an inbuilt Ethernet and USB3 ports for external connectivity. Radio Modems can support bit rate up to 80 Mbit/sec with variable uplink and downlink capacity on demand [22]. Radio Modem is Time Division Duplex (TDD) with variable uplink and downlink frame size with dedicated time slots for broadcasting, initial user connectivity, and synchronization. For the first experimentation, we configure the modem to operate at carrier frequency of 2.5 GHz with useful bandwidth of 4 MHz and data rate of 80 Mbit/sec. Radio Modem exposes socket based application programming interface (API) through which external devices can send/receive both user and control data called as Low Level Radio Data (LLR-D) and Low Level Radio Control (LLR-C) messages.

As shown in Fig. 2, in order to enable Radio Modem to integrate with our SDN framework, we have defined a SDN Agent located in external CPU boards (e.g. Raspberry Pi) implemented using Radio Modem APIs. SDN Agent consists of two sub-functional modules such as MCMP Agent (MCMP-A) for communication with controller for mobility control (e.g. user modem attachment and handover) and socket for binding virtual ports (vPorts) of OVS to Radio Modem for sending/receiving LLR messages using LLR APIs. OVS in IM dynamically creates and deletes vPorts upon receiving instructions via OVSDB protocol in SDN controller during UM attachment and handover procedures. OVS in IM allocates one vPort per UM as shown in Fig. 3. On the UM side, each OVS has one vPort statically created to distribute data over the connected LAN. OpenFlow control messages between UM and SDN controller are communicated via IM.

C. OpenFlow for QoS-aware Service Scheduling

The main objective of integrating OVS with Radio Modem is to have QoS aware flow scheduling using OpenFlow priority queue feature as shown in Fig. 3. Hence, traffic flows belong to different services are scheduled inside the OVS before packets arrive at Radio Modem for sending over the air. Service scheduling application on the top of SDN controller completely manages packet scheduling inside OVS using OpenFlow and OVSDB plugins.

As explained in Fig. 3, the purpose of using OpenFlow protocol in our architecture is to enforce networking rules by assigning flows to traffic shaping queues behind each outport (queue/service) inside OVS. OpenFlow starting from version 1.2 supports QoS such as "set-queue" action to forward packets to each queue based on their predefined rules, query queue statistics from the controller for NB applications to take scheduling decisions dynamically, etc [18]. Although the controller can assign flows to queues and query queue statistics using OpenFlow protocol, it cannot perform certain configuration related actions such as creating queues and

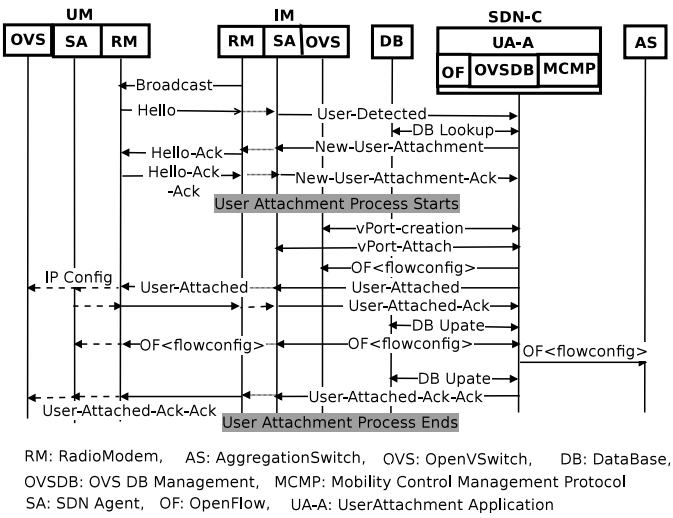


Fig. 4: User-Attachment Scenario.

modifying queue parameters (e.g. maximum and minimum data rate for each queue). In fact, configuration protocols for OpenFlow devices such as OpenFlow Management and Configuration protocol (OF-CONFIG) or OVSDB were introduced in order to perform those missing configuration features from OpenFlow [19]. Even though OVSDB protocol was not supported in any early SDN controllers, ODL controller from Linux foundation exclusively supports OVSDB protocol [8].

D. Use Cases

We are interested in three main procedures related to the mobility management and QoS aware flow based service scheduling namely UM Attachment, UM Handover, and Service Scheduling. In all of them, it is to be considered that all the OVS switches are assigned static IP addresses for controller communication.

1) User Modem Attachment Procedure: This procedure enables the User Modem to register to the network during the initial power on as shown in Fig. 4. All the IMs always broadcast their availability and the location of their downlink *Ranging* slot (a dedicated slot in the frame for connection request to be sent by UM). IMs send broadcast message during dedicated uplink *Synch* slot known by UM. The UM received the broadcast message and sends *Hello* message to IM, which includes its identity and received power level from the IM. On the reception of *Hello* message from UM, IM sends *User-Detected* message via SDN-Agent (SA) using MCMP protocol to UM Attachment application in the NB of SDN Controller (SDN-C). This message includes the information received from UM such as identity and power level. After receiving this message, UM Attachment application executes lookup process in the DB to verify if the UM already exists in the network. If the UM is a new user in the network, the application sends back *User-Attachment* message to IM using MCMP protocol. Then, the IM sends back *Hello-Ack* message to UM. UM then acknowledges IM by sending back *Hello-Ack-Ack* message.

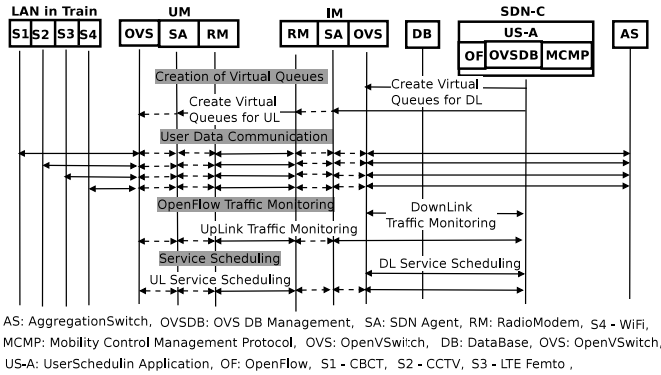


Fig. 5: Service Scheduling Procedure.

Upon receiving this message, IM sends *User-Attachment-Ack* message to UM Attachment application in the SDN-C. Now the application begins the User-Attachment process by creating a new entry for UM in IM's table in the DB. Using OVSDB protocol, application creates vPort for new UM in the IM's OpenVwsitch (OVS) and binds this port to Radio Modem (RM) using dynamic socket connection process by sending *vPort-creation* message to OVS and *vPort-Attach* message to SDN-Agent. After the vPort is established for the UM, UM Attachment application using OpenFlow protocol sends OF<*flowconfig*> message to OVS in IM to configure initial flow setup (DHCP discovery messages) for attached UM traffic to reach the ground network, it is necessary since all the devices connected in the train LAN network needs to get an IP address from the DHCP server located at the edge of the network for end-to-end IP connectivity.

After initial flow configuration for attached UM, UM Attachment application sends *User-Attached* message to the UM including the controller's IP address in order to configure OVS to establish connection with the SDN controller. After the successful configuration, UM sends *User-Attached-Ack* message to IM and UM Attachment application in the controller. Then, the application updates the DB with the new user attachment details. Finally, the application sends OF<*flowconfig*> message to OVS in UM and also to Aggregation Switch (AS) to create flow entry in order to support initial DHCP discovery related traffic to reach the ground network via IM. This completes the UM Attachment procedure and now all the devices in the train starts to discover their services and establish connections via IM. Once all the devices discover their IP addresses, AS needs to have flow rule based on source destination IP addresses so as to route UM traffic to the respective IM. This will be automatically done by routing applications running in most of the SDN/OpenFlow controllers using OpenFlow related services such as topology and switch manager.

2) Service Scheduling Procedure: The Service Scheduling procedure follows immediately after a successful UM Attachment procedure. Service scheduling is required to prioritize certain services especially those related to train control and surveillance systems. It is initiated by Service Scheduling application in the controller. We use priority queuing based flow scheduling feature provided by OF as shown in Fig. 5. The *Service Scheduling* application sends OVSDB messages (ovs-vsctl) to IM OVS and UM OVS to create a number of virtual queues for scheduling both downstream and upstream traffic to and from UM. Service Scheduling application monitors upstream and downstream traffic in the network using OpenFlow traffic monitoring features such as statistics per port, queue and flow in each OVS. The application attaches traffic flow of each service to different virtual queues and also configures maximum and minimum data rate for each virtual queue depending on their priority level. In our design as shown in Fig. 3 we have four different traffic flows. The priority decision and the maximum and minimum data rate for each queue completely depend on the algorithm implemented inside the Service Scheduling application. This procedure repeats periodically as defined in the service scheduling algorithm.

3) User Modem Handover Procedure: Handover procedure follows after UM Attachment and Service Scheduling procedures when the train is moving into the signal coverage of another IM, we name it as IM-2. As shown in Fig. 6 , this procedure follows the same message sequence as User Modem Attachment procedure until the NB *User Modem Handover* application begins the Handover validation process. The User Modem Handover application depends on its handover criteria (power level experienced by UM, load, etc) to decide if the UM has to be attached to IM-2. In case the application decides to execute handover, it sends *User-Attachment* message to IM confirming the handover decision. Then, IM-2 acknowledges UM by sending *Hello-Ack* message. On the reception of hello acknowledgment, UM follows the handover procedure by sending back *HelloAck-Ack* to IM-2. Then, the IM-2 proceeds the handover procedure by sending *User-Attachment-Ack* message to User Modem Handover application. After receiving acknowledgment from IM-2, the handover application creates vPort in the IM-2 OVS for the UM that is under handover procedure by sending *vPort-creation* message to OVS and *vPort-Attach* message to SDN-Agent. In order to avoid packet drops due to handover, application queries the DB for flow details corresponding to existing services in the UM and sets up flow in IM-2's OVS by sending OF<*flowconfig*> message. Finally, the application sends *User-Attached* message to IM-2 and then the IM-2 forwards it to UM. Once the User Modem Handover application receives the *User-Attached-Ack* message from the UM, it modifies the flow rules in Aggregation Switch for all services in UM to be routed via IM-2 by sending OF<*flowmod*> message. It removes the vPort for UM in IM-1 by sending *vPort-Removal* message to IM-1's OVS. This completes the handover procedure and the UM is now served by IM-2.

IV. Architecture Validation

In order to perform initial validation of our architecture, we have implemented a software prototype as described in Fig. 7, using ODL controller, MongoDB, Open vSwitches, and Wireless Modem emulator from Simpulse [22]. For the

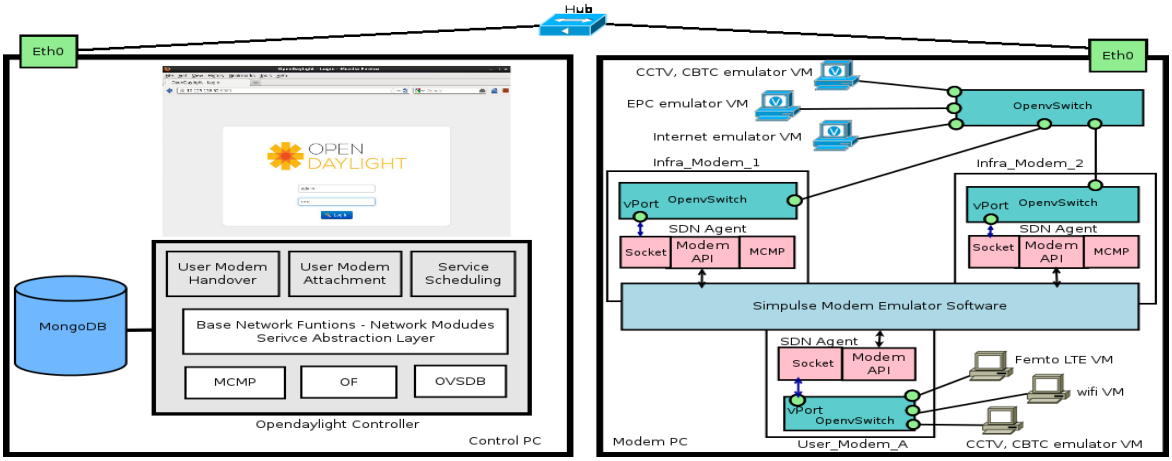


Fig. 7: Software Prototype Architecture.

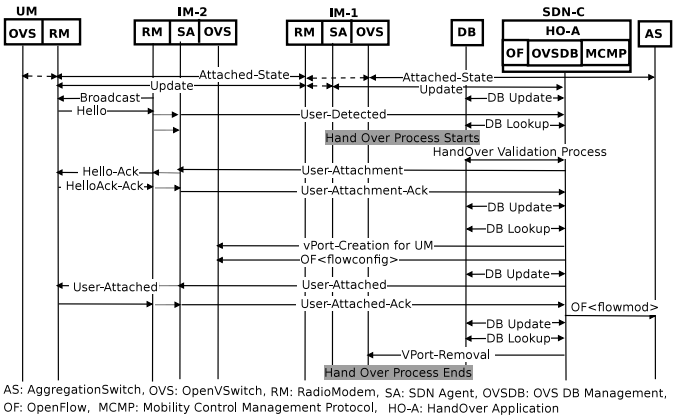
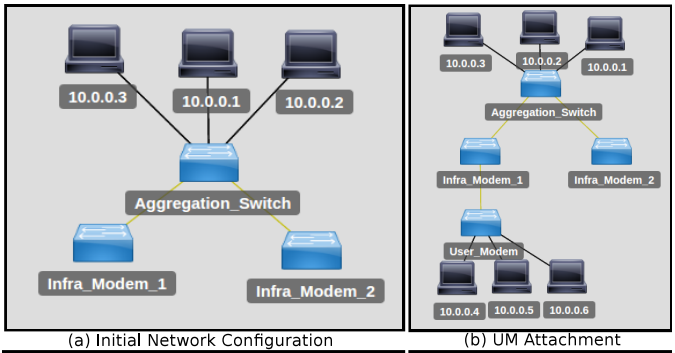


Fig. 6: User Modem Handover Procedure.

experimentation, We used two PCs having Linux Ubuntu distribution with low latency kernel version 3.13 running on Intel i5 at 3.2 GHz and 8GB RAM. We deployed control-plane components such as SDN controller, applications and DBMS in one dedicated PC, and data-plane nodes such as modems and Open vSwitches in another PC as shown in Fig. 7. We have chosen ODL since it supports both OF and OVSDB protocols together and provides flexible framework to integrate any new protocol plugins. The choice of MongoDB is due to high performance compared to SQL based DBMS systems [10]. We have implemented MCMP plugin in the SB of ODL to exchange LLR messages between Infra Modems and the mobility control applications in the NB of the controller. We also integrated new service modules for MCMP management in the controller with necessary modifications to both Service Abstraction Layer (SAL) and NB API layer to provide necessary interfaces for NB applications. Our modem emulator software is able to emulate any number of Infra and User modems. We developed SDN-Agent that uses Modem APIs to interface Radio Modems to SDN environment (SDN controller, OVS).

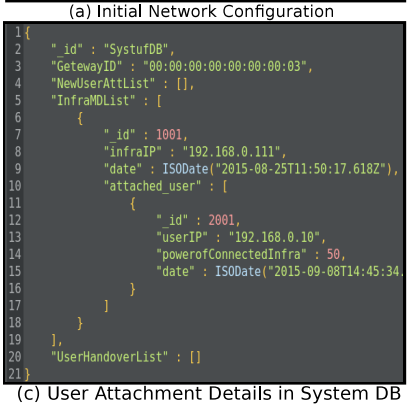
In our particular scenarios, we emulated two Infra Modems and one User Modem as in Fig.7, to demonstrate User Modem Attachment and User Modem Handover procedures. We have deployed three Open vSwitches and integrated them with emulated modems using our SDN-Agent module. Open vSwitch deployed at the edge of the network acts as an Aggregation Switch (AS) which connects Mobile Backhaul network and networks of service providers in the ground. In order to emulate end-to-end traffic, we have attached virtual machines to the AS and to the User Modem. As a first step towards validation of the framework, we deployed User Modem Attachment and User Modem Handover applications in the NB of the controller. All the information regarding UM Attachment and UM Handover such as the ID of UM and the received power level are stored in the DB and the NB applications can access them through APIs in the NB of the controller.

According to our scenario, the initial network configuration looks as Fig. 8(a), where two Infra Modems are connected to Aggregation Switch and attached to the controller. Also three Virtual Machines (VMs) are connected to Aggregation Switch to represent 3 different services such as Internet, LTE core network, and Transport Control Systems (CCTV, CBTC). Now, the IMs started to broadcast and listen for new UMs to be attached. As defined in our validation scenario, after five seconds the emulated UM started to send attachment request to the Infra Modem. As described in Fig. 8(b), the UM attachment procedure is completed in 3 ms and we validated it by successfully sending ICMP messages between VMs connected to the UM and to the Aggregation Switch. The user attachment details in system DB is shown in Fig. 8(c). After 1 minute, the handover procedure begins when the UM started to attach with the IM-2. The handover application begins to execute the procedure when the power received by UM from IM-2 is higher than that from IM-1. As shown in Fig. 8(d), the handover procedure is completed in 6 ms. The ICMP messages between VMs connected to UM and to Aggregation Switch continuously flows but through IM-2 instead of IM-1 after



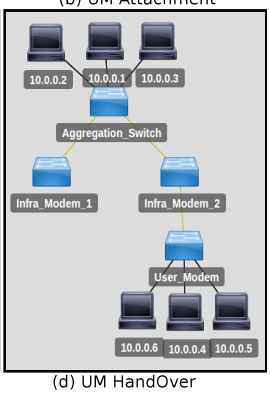
(a) Initial Network Configuration

(b) UM Attachment



```
{
    "_id" : "SystufDB",
    "GeteWayID" : "00:00:00:00:00:00:00:03",
    "NewUserAttList" : [],
    "InfraMDList" : [
        {
            "_id" : 1001,
            "infraIP" : "192.168.0.111",
            "date" : ISODate("2015-08-25T11:50:17.618Z"),
            "attached_user" : [
                {
                    "_id" : 2001,
                    "userIP" : "192.168.0.10",
                    "powerofConnectedInfra" : 50,
                    "date" : ISODate("2015-09-08T14:45:34.
                }
            ]
        }
    ],
    "UserHandoverList" : []
}
```

(c) User Attachment Details in System DB



(d) UM HandOver

Fig. 8: Validation Scenario.

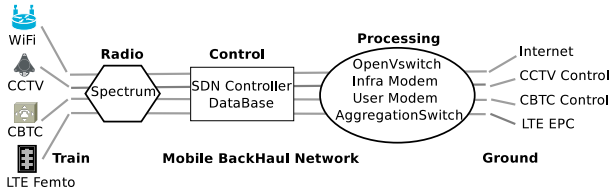


Fig. 9: Network Slices in Mobile Backhaul.

handover. Hence by this process we successfully validated both the attachment and handover procedures.

V. Conclusion & Future Work

Although there are various solutions to provide high speed communication services for transport systems, they cannot offer guaranteed QoS for certain services in dynamically varying traffic conditions. In this work, we introduced the concept of SDN in the train to ground communication systems in order to bring programmability in the mobile backhaul. As depicted in Fig. 9, the primary goal of this work is to slice end-to-end backhaul network resources among different services based on their level of priority and QoS requirements. The proposed framework enables the delivery of *network as a service* on-demand and tailors resources accordingly. The main advantages from our SDN based framework especially dynamic programmability and reconfigurability enable mobile backhaul to have mobility management driven directly from locally deployed SDN controller rather than from dedicated mobility management entity deployed in core networks as in traditional wireless systems. Taking the advantages from OpenFlow and OVSDB protocols, we are also able to efficiently manage scarce radio resources using pre-scheduling via Open vSwitches integrated with Radio Modems.

As an initial step, we have implemented our architecture in a software prototype and validated scenarios related to mobility management. The complete demo on a standard hardware platform with all the described use cases and several other potential use cases such as interference management between Infra Modems is an ongoing work and planned for future validation of our architecture and system design.

Acknowledgment

This research was performed under project SYSTUF, which received funding from French Ministry of Industry in the framework the AMI ITS program.

References

[1] A. Bertout and E. Bernard, "Next generation of railways and metros wireless communication systems," *ASPECT, Institution of Railway Signal Engineers*, 2012.
[2] Alcatel Lucent LTE for Metro Railway Operations White Paper. [Online] http://www.tmcnet.com/tmc/whitepapers/documents/whitepapers/2013/8272-alcatel-lucent-lte-metro-railway-operations.pdf.
[3] 3rd Generation Partnership Project. http://www.3gpp.org.
[4] D. T. Fokum, S. Member, and V. S. Frost, "A survey on methods for broadband Internet access on trains," *IEEE Communications Surveys and Tutorials*, vol. 12, no. 2, pp. 171–185, 2010.
[5] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling innovation in campus networks," *SIGCOMM Comput. Commun. Rev.*, Mar. 2008.
[6] L. Madhusanka, G. Andrei, and Y. Mika, *Software Defined Mobile Networks (SDMN): Beyond LTE Network Architecture*. Wiley, 2015.
[7] P. Costa, M. Migliavacca, P. Pietzuch, and A. L. Wolf, "NaaS: Network-as-a-service in the cloud," in *USENIX Conf. on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and Services*, 2012.
[8] OpenDayLight. https://www.opendaylight.org.
[9] Open virtual switch. [Online] http://openvswitch.org.
[10] MongoDB White Paper: Performance Best Practices for MongoDB. [Online] http://s3.amazonaws.com/info-mongodb-com/MongoDB-Performance-Best-Practices.pdf.
[11] S. Itziar, C. Roberto, and P. Asier, *Wireless Technologies in the Railway: Train-to-Earth Wireless Communications*. InTech, 2012.
[12] L. Adrian, K. Anisha, and R. Byrav, "Network innovation using OpenFlow: A survey," *IEEE Communications Surveys and Tutorials*, 2013.
[13] M. Bansal, J. Mehlman, S. Katti, and P. Levis, "OpenRadio: a programmable wireless dataplane," in *ACM HotSDN*, 2012, pp. 109–114.
[14] A. Gudipati, D. Perry, L. E. Li, and S. Katti, "SoftRAN: software defined radio access network." in *ACM HotSDN*, 2013, pp. 25–30.
[15] A. Gopalasingham, L. Roullet, N. Trabelsi, C. S. Chen, A. Hebbar, and E. Bizouarn, "Generalized software defined network platform for radio access networks," in *IEEE Consumer Commun. & Networking Conf.*, 2016.
[16] D. Boviz, A. Gopalasingham, , C. S. Chen, and L. Roullet, "Physical layer split for user selective uplink joint reception in SDN enabled Cloud-RAN," in *Australian Communications Theory Workshop*, 2016.
[17] D. Bercovich, L. M. Contreras, Y. Haddad, A. Adam, and C. J. Bernardos, "Software-defined wireless transport networks for flexible mobile backhaul in 5G systems," *Mobile Networks and Applications*, vol. 20, no. 6, pp. 793–801, 2015.
[18] S. Balzs, G. Andrs, N. Felicin, C. Jnos, K. Krisztin, N. Barnabs, and V. Gbor, "On QoS support to Ofelia and OpenFlow," in *European Workshop on Software Defined Networking (EWSDN)*, 2012.
[19] P. David, G. Joao, S. Bruno, C. Luis, S. Paulo, S. Sachin, and S. Dimitri, "The QueuePusher: Enabling queue management in OpenFlow," in *European Workshop on Software Defined Networking (EWSDN)*, 2014.
[20] ZeroMQ Distributed Messaging. [Online] http://zeromq.org.
[21] PULSAR Board. [Online] http://www.simpulse-dsp.com/offer/tools/pulsar-board.
[22] Software Defined Radio Modem. [Online]http://www.simpulse-dsp.com.